1. Explain why selenium is important in web scraping?
   Ans:. Selenium is an automation testing framework for web applications/websites which can also control the browser to navigate the website just like a human. Selenium uses a web-driver package that can take control of the browser and mimic user-oriented actions to trigger desired events. This guide will explain the process of building a web scraping program that will scrape data and download files from Google Shopping Insights.Setup
   Selenium : To download selenium package, execute the below pip command in terminal:
   Pip install selenium
   Python
   Selenium Drivers: Web drivers enable python to control the browser via OS-level interactions. Web drivers use the browser's built-in support for the automation process so, in order to control the browser, the web-driver must be installed and should be accessible via the PATH variable of the operating system (only required in case of manual installation).
   Download the drivers from official site for Chrome, Firefox, and Edge. Opera drivers can also be downloaded from the Opera Chromium project hosted on Github.Data Extraction
   Let's get started by searching a product and downloading the CSV file(s) with the following steps:

   Import Dependencies and Create Driver Instance: The initial step is to create an object of webdriver for particular browser by importing it from selenium module as:
   From selenium import webdriver # Import module
   From selenium.webdriver.common.keys import Keys # For keyboard keys
   Import time # Waiting function
   URL = 'https://shopping.thinkwithgoogle.com' # Define URL
   Browser = webdriver.Safari() # Create driver object means open the browser
   Python
   By default, the automation control is disabled in safari and it needs to be enabled for automation environment otherwise it will raise SessionNotCreatedException. So, enable the Develop option under the advanced settings in Safari preferences.

2. What's the difference between scraping images and scraping websites?

Use an example to demonstrate your point.

Ans:. What is image scraping?

Image scraping is a subset of the web scraping technology. While web scraping deals with all forms of web data extraction, image scraping only focuses on the media side – images, videos, audio, and so on.

How is image scraping done?

While there are several tools and techniques available to extract images from websites, we'll take a look at two solutions provided by Grepsr — Grepsr Concierge and Grepsr Browser Extensions — in this article.

Via Grepsr Concierge

Grepsr's Concierge service is the perfect solution for bulk image extraction requirements — such as multiple image URLs for an item, or extracting images as JPG or PNG files, compressing them into zip files, applying a certain file naming format and so on.

Once we receive your project details, our team of expert engineers get to work setting up the project tailored specifically to your requirements.Imper

What is web scraping

Web scraping is the process of using bots to extract content and data from a website.

Unlike screen scraping, which only copies pixels displayed onscreen, web scraping extracts underlying HTML code and, with it, data stored in a database. The scraper can then replicate entire website content elsewhere.

Web scraping is used in a variety of digital businesses that rely on data harvesting. Legitimate use cases include:

Search engine bots crawling a site, analyzing its content and then ranking it.

Price comparison sites deploying bots to auto-fetch prices and product descriptions for allied seller websites.

Market research companies using scrapers to pull data from forums and social media (e.g., for sentiment analysis).

Web scraping is also used for illegal purposes, including the undercutting of prices and the theft of copyrighted content. An online entity targeted by a scraper can suffer severe financial losses, especially if it's a business strongly relying on competitive pricing models or deals in content distribution.

Scraper tools and bots

Web scraping tools are software (i.e., bots) programmed to sift through databases and extract information. A variety of bot types are used, many being fully customizable to:

Recognize unique HTML site structures

3. Explain how MongoDB indexes data.?

Ans: Indexes support the efficient execution of queries in MongoDB. Without indexes, MongoDB must perform a collection scan, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.Indexes are special data structures [1] that store a small portion of the collection's data set in an easy to traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field. The ordering of the index entries supports efficient equality matches and range-based query operations. In addition,

MongoDB can return sorted results by using the ordering in the index. The following diagram illustrates a query that selects and orders the matching documents using an index:Diagram of a query that uses an index to select and return sorted results. The index stores ``score`` values in ascending order. MongoDB can traverse the index in either ascending or descending order to return sorted results. Fundamentally, indexes in MongoDB are similar to indexes in other database systems. MongoDB defines indexes at the collection level and supports indexes on any field or sub-field of the documents in a MongoDB collection.Default _id Index MongoDB creates a unique index on the _id field during the creation of a collection. The _id index prevents clients from inserting two documents with the same value for the _id field. You cannot drop this index on the _id field.Index Names The default name for an index is the concatenation of the indexed keys and each key's direction in the index (i.e. 1 or -1) using underscores as a separator. For example, an index created on { item : 1, quantity: -1 } has the name item_1_quantity_-1.You can create indexes with a custom name, such as one that is more human-readable than the default. For example, consider an application that frequently queries the products collection to populate data on existing inventory. The following createIndex() method creates an index on item and quantity named query for inventory:.products.createIndex({ item: 1, quantity: -1 } , { name: "query for inventory" }You can view index names using the db.collection.getIndexes() method. You cannot rename an index once created. Instead, you must drop and re-create the index with a new name.Index TypesMongoDB provides a number of different index types to support specific types of data and queries. Single Field In addition to the MongoDB-defined _id index, MongoDB supports the creation of user-defined ascending/descending indexes on a single field of a document. Multikey Index MongoDB uses multikey indexes to index the content stored in arrays. If you index a field that holds an array value, MongoDB creates separate index entries for every element of the array. These multikey indexes allow queries to select documents that contain arrays by matching on element or elements of the arrays. MongoDB automatically determines whether to create a multikey index if the indexed field contains an array value; you do not need to explicitly specify the multikey type.

4. What is the significance of the SET modifier?

Ans:. Set modifiers

Set expressions are used to define the scope of a calculation. The central part of the set expression is the set modifier that specifies a selection. This is used to modify the user selection, or the selection in the set identifier, and the result defines a new scope for the calculation.

The set modifier consists of one or more field names, each followed by a selection that should be made on the field. The modifier is enclosed by angled brackets: < >

For example:

Sum ({$<Year = {2015}>} Sales)

Count ({1<Country = {Germany}>} distinct OrderID)

Sum ({$<Year = {2015}, Country = {Germany}>} Sales)

Element sets

An element set can be defined using the following:

A list of values

A search

A reference to another field

A set function

If the element set definition is omitted, the set modifier will clear any selection in this field. For example:

Sum({$<Year = >} Sales)

Examples: Chart expressions for set modifiers based on element sets

Examples – chart expressions

Listed values

The most common example of an element set is one that is based on a list of field values enclosed in curly brackets. For example:

{$<Country = {Canada, Germany, Singapore}>}

{$<Year = {2015, 2016}>}

The inner curly brackets define the element set. The individual values are separated by commas.

Quotes and case sensitivity

If the values contain blanks or special characters, the values need to be quoted. Single quotes will be a literal, case-sensitive match with a single field value. Double quotes imply a case-insensitive match with one or several field values. For example:

<Country = {'New Zealand'}>

Matches New Zealand only.

<Country = {"New Zealand"}>

Matches New Zealand, NEW ZEALAND, and new zealand.

Dates must be enclosed in quotes and use the date format of the field in question.

Double quotes can be substituted by square brackets or by grave accents.

5. Explain the MongoDB aggregation framework.?

Ans:. The aggregation framework is a set of analytics tools within MongoDB that allows us to run various types of reports or analysis on documents in one or more collections. Based on the idea of a pipeline. We take input from a MongoDB collection and pass the documents from that collection

through one or more stages, each of which performs a different operation on it's inputs. Each stage takes as input whatever the stage before it produced as output. And the inputs and outputs for all stages are a stream of documents. Each stage has a specific job that it does. It's expecting a specific form of document and produces a specific output, which is itself a stream of documents. At the end of the pipeline, we get access to the output.

Aggregation framework stage

An individual stage is a data processing unit. Each stage takes as input a stream of documents one at a time, processes each document one at a time and produces the output stream of documents. Again, one at a time. Each stage provide a set of knobs or tunables that we can control to parameterize the stage to perform whatever task we're interested in doing. So a stage performs a generic task – a general purpose task of some kind and parameterize the stage for the particular set of documents that we're working with. And exactly what we would like that stage to do with those documents. These tunables typically take the form of operators that we can supply that will modify fields, perform arithmetic operations, reshape documents or do some sort of accumulation task as well as a veriety of other things. Often times, it the case that we'll want to include the same type of stage multiple times within a single pipeline.

Same type of stage multiple times within a single pipeline

e.g. We may wish to perform an initial filter so that we don't have to pass the entire collection into our pipeline. But, then later on, following some additional processing, want to filter once again using a different set of criteria. So, to recap, pipeline works with a MongoDB collection. They're composed of stages, each of which does a different data processing task on it's input and produces documents as output to be passed to the next stage. And finally at the end of the pipeline output is produced that we can then do something within our application. In many cases, it's necessary to include the same type of stage, multiple times within an individual pipeline.